NSS Port to SPEEDES

"Much Faster Than Real Time Strike Plan Evaluation"

Phase 1: Data Distribution Management Design Review

Dr. Jeff Steinman
Metron Inc.
(619) 792-8904
steinman@ca.metsci.com

Dr. Bill Stevens
Metron Inc.
(619) 792-8904
stevens@ca.metsci.com

KEY TEAM MEMBERS

- NRL
  - Jim Hofmann
  - Dr. Bill Smith
  - Dr. Roger Hillson
- Metron
  - Dr. Bill Stevens
  - Dr. Jeff Steinman
  - Jeff Jones
  - Jeff Monroe
  - Gary Blank
  - Jacob Burckhardt
  - Colleen Gagnon

OUTLINE

- Overview
  - APPEX, NSS, SPEEDES, HLA
  - Schedule
  - Objectives
  - Approach
- Technical Background
  - Architecture
  - SPEEDES - Software Impact on NSS
  - Data Distribution Management
  - HLA Interfaces
  - Miscellaneous Ongoing Tasks
- Summary

OVERVIEW

- *Common HPC Software Support Initiative*
  - *(CHSSI) sponsored by HPCMO*
  - *Collaboration between NRL and Metron*
  - *Currently in Phase 1 of a three phase effort*
- *Rehost NSS using SPEEDES to execute on high-performance parallel machines to support much faster than real time evaluation of strike plans generated by APPEX*
  - *Exemplar, Intel Paragon, IBM SP-2, Networks of workstations*
- *Develop multipurpose data distribution software*
  - *Critical starting point for this effort*
  - *Potential reuse in other parallel simulations*
  - *Potential reuse for HLA*

APPEX

- Advanced Power Projection Planning and Execution Project
  - Developed by NRL, Metron, & Autometric
  - Used by carrier air wing
- Force-level strike planning software prototype tool
  - Assignment, routing, weaponeering, scheduling, and timeline generation
- Requires a much faster than real-time simulator for evaluation and previewing results
  - Parallel NSS on HPC using SPEEDES

NSS

- Naval Simulation System
  - Resource Sponsors: CNO, N6, and N8
  - Dev. Team: Metron (Sim. Engine), SAIC (HCI), ARL/UT (Data)
  - Initial User Sites: N812, CINCPACFLT, PAX River, APL
  - Program Manager: Dr. Les Parish, SPAWAR 312-6
- Object-oriented discrete-event simulation (C++)
  - Navy and Joint Service Assessments
    - JMA assessments, investment balance review, force mix analysis
  - Operational Planning, Course of action, Devel./Assessment
    - TAMPS integration, JMCIS integration underway
  - Training (JTFEX participation)
- NSS already supports strike plan evaluations (for TAMPS) & is a member of the JTFp HLA federation

SPEEDES

- Synchronous Parallel Environment for Emulation and Discrete-Event Simulation
  - Government owned software (NASA/JPL) with two patents pending and over 20 publications
  - Used by: JPL, Metron, NRaD, MITRE, Aerospace, Northrop, AEgis, ICASE, Dartmouth
- Flexible parallel simulation framework
  - Scalable object-oriented software engineering in C++
    - Active event objects / passive simulation objects
  - Multiple synchronization strategies (TW, FTB, BTB, BTW)
    - Breathing Time Warp provides flow control for optimistic sims.
  - Portable/optimizable to virtually all hardware platforms
- Designed to support the needs of military simulations

HLA

- *High Level Architecture*
  - *Sponsored by DMSO*
  - *Virtually all future military simulations must become HLA compliant*
- *Provides interoperability between distributed simulations (open system approach)*
  - *Defined as: Interfaces, Object Models, Rules*
  - *Different time schemes: DIS, ALSP*
  - *Heterogeneous computers (different data formats)*
  - *New systems: JWARS, JSIMS*
- *SPEEDES-based DDM and TM reused for HLA*
  - *Future HLA support for HPC*

SCHEDULE

- Phase 1 - Now through June 97'
  - Complete Data Distribution Management effort
    - Design will support HLA functionality
  - Prototype experimental HLA interfaces
  - Demonstrate performance on Convex Machine
- Phase 2 - June 97' through June 98'
  - Begin rehosting NSS software in SPEEDES
  - Demonstrate portability on multiple platforms
- Phase 3 - June 98' through December 98'
  - Complete the rehosting of NSS software
  - Integrate with APPEX
  - Demonstrate much faster than real time strike plan evaluation

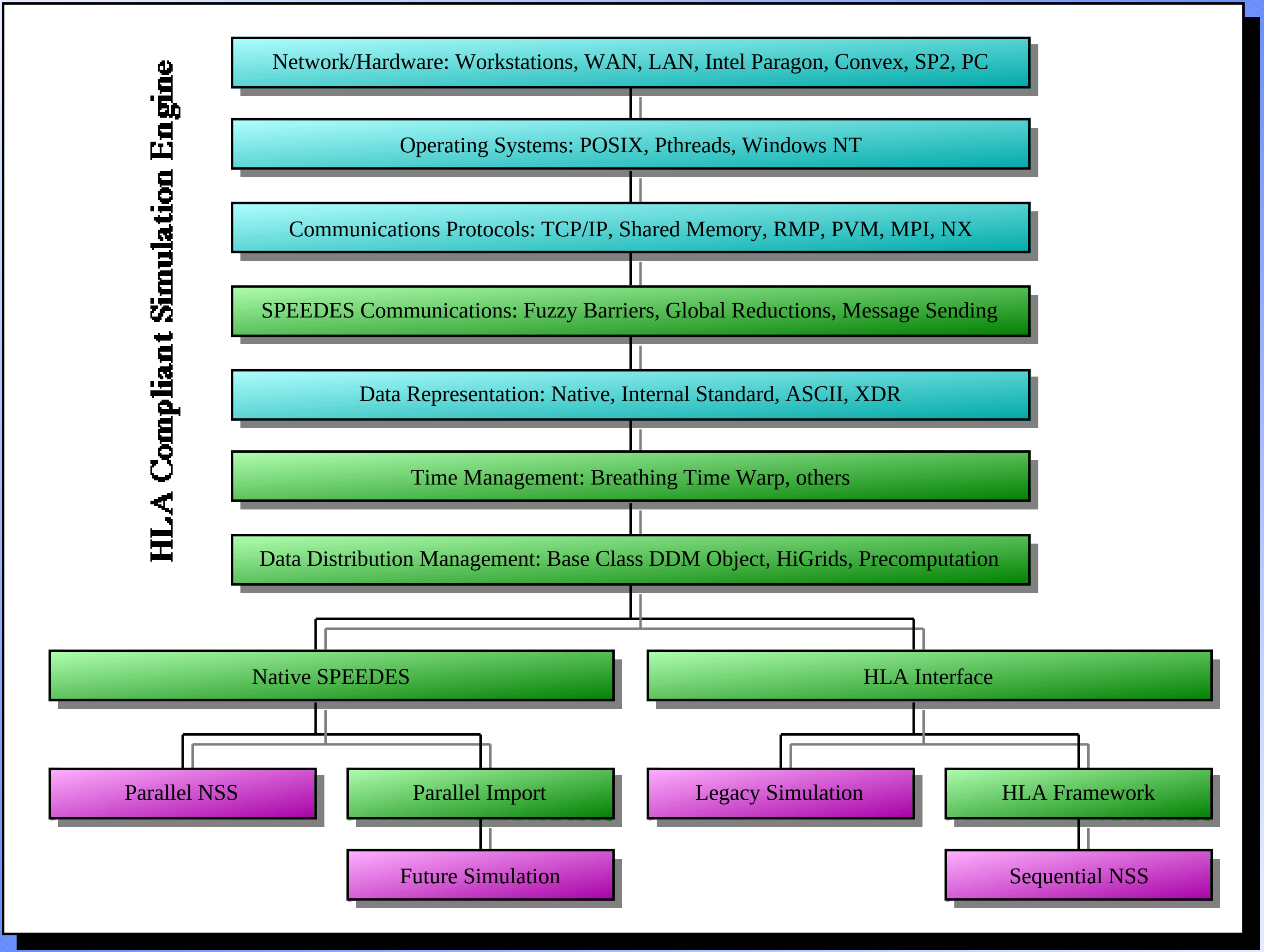
OBJECTIVES

- *Meeting these objectives is critical to our success!*
  - *Functionality*
    - *Must support current NSS functionality & integrate with APPEX*
  - *Performance*
    - *Must execute fast in parallel on a wide variety of machines without sacrificing current NSS sequential performance*
    - *Must solve the data distribution problem in a scalable manner*
      - *Data Distribution - how to efficiently disseminate information about remote objects in a distributed environment*
      - *Scalability - Doubling the number of machines allows the problem size to be doubled with the same run-time performance*
  - *Software Development*
    - *Must minimize the amount of new software required*
    - *Must develop a maintainable software design*
    - *Must not be overly complex due to parallel considerations*

APPROACH

- Approach to meeting critical objectives
  - Functionality
    - – Rehost NSS to run in SPEEDES & integrate with APPEX
  - Performance
    - – SPEEDES is portable to different machines and eliminates parallel overheads when running sequentially
    - – Objects distributed to different processors (scatter decomposition will be used initially)
    - – Scalable data distribution management design
      - Addresses CPU usage, messages, and memory
  - Software Development
    - – SPEEDES provides automatic support for parallel simulation
    - – Data distribution management simplifies rest of port
    - – Standard software development practices and metrics

ARCHITECTURE


HLA Compliant Simulation Engine
Network/Hardware: Workstations, WAN, LAN, Intel Paragon, Convex, SP2, PC
Operating Systems: POSIX, Pthreads, Windows NT
Communications Protocols: TCP/IP, Shared Memory, RMP, PVM, MPI, NX
SPEEDES Communications: Fuzzy Barriers, Global Reductions, Message Sending
Data Representation: Native, Internal Standard, ASCII, XDR
Time Management: Breathing Time Warp, others
Data Distribution Management: Base Class DDM Object, HiGrids, Precomputation
Native SPEEDES
HLA Interface
Parallel NSS
Parallel Import
Legacy Simulation
HLA Framework
Future Simulation
Sequential NSS

ARCHITECTURE (CONT.)

- Network/Hardware
  - Defining the hardware that is of interest to this project
  - Currently existing
    - Workstations (HP, SUN, SGI)
      - Shared memory
      - TCP/IP
    - Intel Paragon
    - Convex Exemplar
  - Soon to be supporting
    - SP2
    - Cray T3D
  - Long term planned support
    - WAN (Multicasting)
    - PCs

ARCHITECTURE (CONT.)

- Operating Systems
  - Use only the most commonly supported UNIX services to maintain portability across all platforms
  - Currently existing
    - POSIX
  - Soon to be supporting
    - Pthreads
  - Long term planned support
    - Windows NT

ARCHITECTURE (CONT.)

- Communications Protocols
  - Standard communications protocols that are of interest to this project
  - Currently existing
    - TCP/IP (Internet Standard)
    - Shared Memory (without semaphores)
    - PVM (Parallel Virtual Machine)
    - NX (Paragon)
  - Soon to be supporting
    - MPI (Message Passing Interface)
  - Long term planned support
    - RMP (Reliable Multicast Protocol)

ARCHITECTURE (CONT.)

- *SPEEDES Communications*
  - *Defines/provides set of communications services used by SPEEDES with a common interface*
  - *Services can be optimized for different hardware/networks*
  - *Link procedure*
    - *Application + SPEEDES Comm. Library + SPEEDES Library*
  - *Currently existing*
    - *TCP/IP + Shared Memory (without semaphores)*
    - *PVM (originally done by Aerospace, not used anymore)*
    - *NX (Native port to Intel Paragon by Dartmouth)*
  - *Soon to be supporting*
    - *MPI (work being done by Dartmouth)*
  - *Long term planned support*
    - *RMP + Shared Memory (collaboration with NRaD)*

ARCHITECTURE (CONT.)

- Data Representation
  - Provides interoperability between machines with different data representations (byte ordering, etc.)
  - Can cause excessive overheads, especially if not needed
  - Currently existing
    - Native data representation
    - Formal Data Representation layer not heeded for HPC
  - Soon to be supporting
    - No near-term plans
  - Long term planned support
    - Internal Standard based on XDR and C++
      - XDR (External Data Representation is standard)
      - Automate through operator overloading
      - Can disable for HPC

ARCHITECTURE (CONT.)

- Time Management (TM)
  - SPEEDES supports multiple synchronization protocols & new approaches independent of other layers in the architecture
  - Currently existing
    - Fixed Time Buckets (conservative, requires lookahead)
    - Breathing Time Buckets (optimistic, risk-free message sending)
    - Time Warp (optimistic, risky message sending)
    - Breathing Time Warp (optimistic, flow control for risk & optimism)
    - Hybrid external interactions with GVT barriers for synchronization
  - Soon to be supporting
    - Dynamic load balancing (Dartmouth)
  - Long term planned support
    - Real-time scheduling
    - Threads on shared memory machines

ARCHITECTURE (CONT.)

- Data Distribution Management (DDM)
  - *Based on Parallel Proximity Detection work in 1993*
  - *Based on NSS Precomputation algorithm*
  - *Based on HLA concept*
  - *Currently existing*
    - *Distributed Hierarchical Grids*
  - *Soon to be supporting*
    - *Geographical theater representations*
    - *General object attribute packaging (static and dynamic)*
    - *Distribution list delivery mechanism*
    - *Precomputations*
  - *Long term planned support*
    - *Real-time data distribution with multicast group delivery services*

ARCHITECTURE (CONT.)

- *Native SPEEDES*
  - *Currently existing*
    - *Incremental state saving services (wide variety)*
    - *Lazy cancellation with tolerances*
    - *Active event objects, passive simulation objects*
      - *Application events inherit from base class event object in SPEEDES.*
      - *Event code is invoked in phases through virtual functions by SPEEDES*
      - *Simulation objects are passive (therefor, reusable)*
  - *Soon to be supporting*
    - *Extensible incremental state saving*
      - *Rogue Wave Tools, others...*
  - *Long term planned support*
    - *Rollforward on query stragglers*
    - *Object blocking mechanisms to reduce rollbacks*

ARCHITECTURE (CONT.)

- Parallel Import
  - NRaD - Larry Peterson & Jeffrey Wallace related CHSSI work
  - Currently existing
    - Sequential process model (MODSIM-like)
    - Integrated expert system support (DOME - extension of Prolog)
    - Persistent data storage
    - CASE tools
  - Soon to be supporting
    - Parallel simulation
  - Long term planned support
    - Distributed debugging & profiling tools
    - Fully integrated HLA compliant simulation development package

ARCHITECTURE (CONT.)

- HLA Interfaces
  - Goal is to provide TM & DDM interfaces defined by the HLA Interface Specifications for federates
  - Currently existing
    - RTI developed by MITRE & Lincoln Labs
  - Soon to be supporting
    - Experimental TM and DDM services based on this CHSSI effort
  - Long term planned support
    - Experiments to optimize TM & DDM performance
    - Full featured RTI optimized for HPC
    - Interoperability with other computing platforms

ARCHITECTURE (CONT.)

- *HLA Framework*
  - *Related to, but not funded by this CHSSI task*
  - *Goal is to make HLA integration as transparent as possible*
  - *Currently existing*
    - *NSS/RTI Framework (used in JTFp integration)*
    - *Others...*
  - *Soon to be supporting*
    - *Integration to new C++ interfaces defined by DMSO*
  - *Long term planned support*
    - *Further interoperability between logical and real time federates*

ARCHITECTURE (CONT.)

- *Parallel / Sequential NSS*
  - *Phases 2 & 3 focus on parallel NSS (Phase 1 is the DDM layer)*
  - *Sequential NSS is currently in production mode*
  - *Currently existing*
    - *Sequential NSS is integrated with HLA (JTFp)*
  - *Soon to be supporting*
    - *Near term experiments in sequential NSS using Hierarchical Grids*
    - *Parallel NSS running on HPC using SPEEDES with native HLA compliant engine*
  - *Long term planned support*
    - *Integrate with APPEX for much faster than real time evaluation of strike plans*
    - *Potential future uses of rollback-based interactive simulations*

ARCHITECTURE (CONT.)

- *Future & Legacy Simulations*
  - *Support for HLA compliant legacy simulations on HPC*
  - *Support for new projects using state-of-the-art technologies*
  - *Currently existing*
    - *JTFp (NSS, Eagle, NASM, DEEM, JTF Headquarters)*
    - *Other prototype federations*
  - *Soon to be supporting*
    - *NSS (sequential HLA version) for experimentation*
  - *Long term planned support*
    - *JTFp on HPC as a possible experiment*
    - *CHSSI has identified JSIMS and JWARS as potential future systems for technology transfer*

TECHNICAL BACKGROUND

- SPEEDES - Software Impact on NSS
  - Conservative approaches require limitations on how tightly objects can interact in logical time
  - Optimistic approaches do not impose such limitations
  - SPEEDES supports both approaches, but...
    - Programming rules must be followed very carefully
      - Events can only interact with a single object
      - State-saving rules for supporting rollback
    - Rules have a big impact on software development
    - SPEEDES does not provide automatic data distribution management services
      - Services must be built at the application level
      - Critical starting point for this project

DATA DISTRIBUTION PROBLEM

- *Sequential, non distributed simulations can permit direct data access between simulated entities*
  - *All entities "live" on one machine*
- *However, distributed simulations must closely regulate data access between simulated entities in order to achieve speedup given communication latencies between different processors*
- *Hence, fundamental problem for distributed simulations is how to manage data accesses between simulated entities*
- *This is also a fundamental problem for the DMSO HLA*

DATA DISTRIBUTION APPROACH

- *Each NSS object will inherit from a base class "Data Distribution Management object" which will provide attributes for and receive attributes from other objects*
  - *DDM base class provides APIs for NSS entities*
- *Rest of NSS operations remain unaltered*
- *Results in a general, minimally intrusive mechanism for adding required distributed functionality to sequential simulation objects*
- *Design supports HLA Data Distribution services*
  - *Can provide services for conservative federates*
  - *HLA objects in a federate map (not inherit) to SPEEDES base class DDM objects which manage data distribution*

EXAMPLE: INTEREST SPACES

Example: Range-based Sensing

Rules:

(1) True sensor coverage must be within subscription region

(2) True detectable object must be within update region

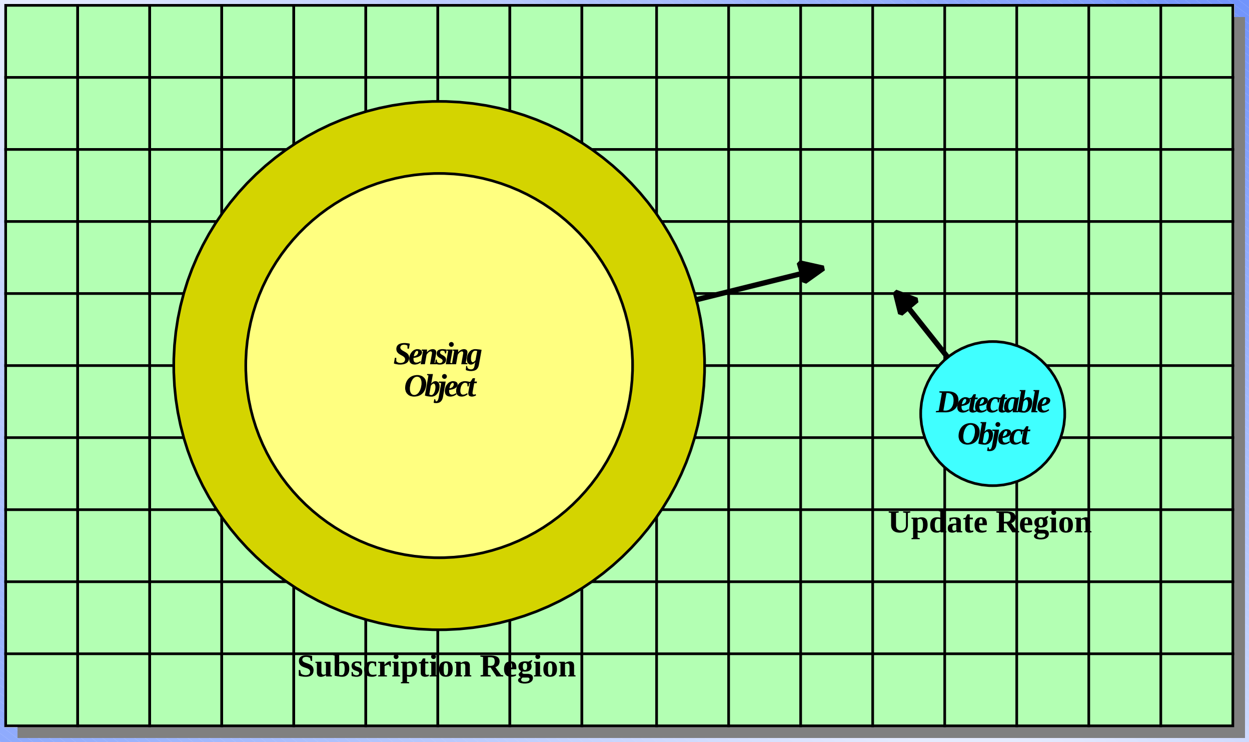
Concept:

(1) δt(i) is chosen to make region modifications a background task for object i

(2) δr(i) = δt(i) Vmax(i) for object i guarantees rules 1 and 2

(3) At time t(i), expand update/ subscription regions for object i by δr(i)

(4) Reschedule next modify region event for time t(i)+δt(i)



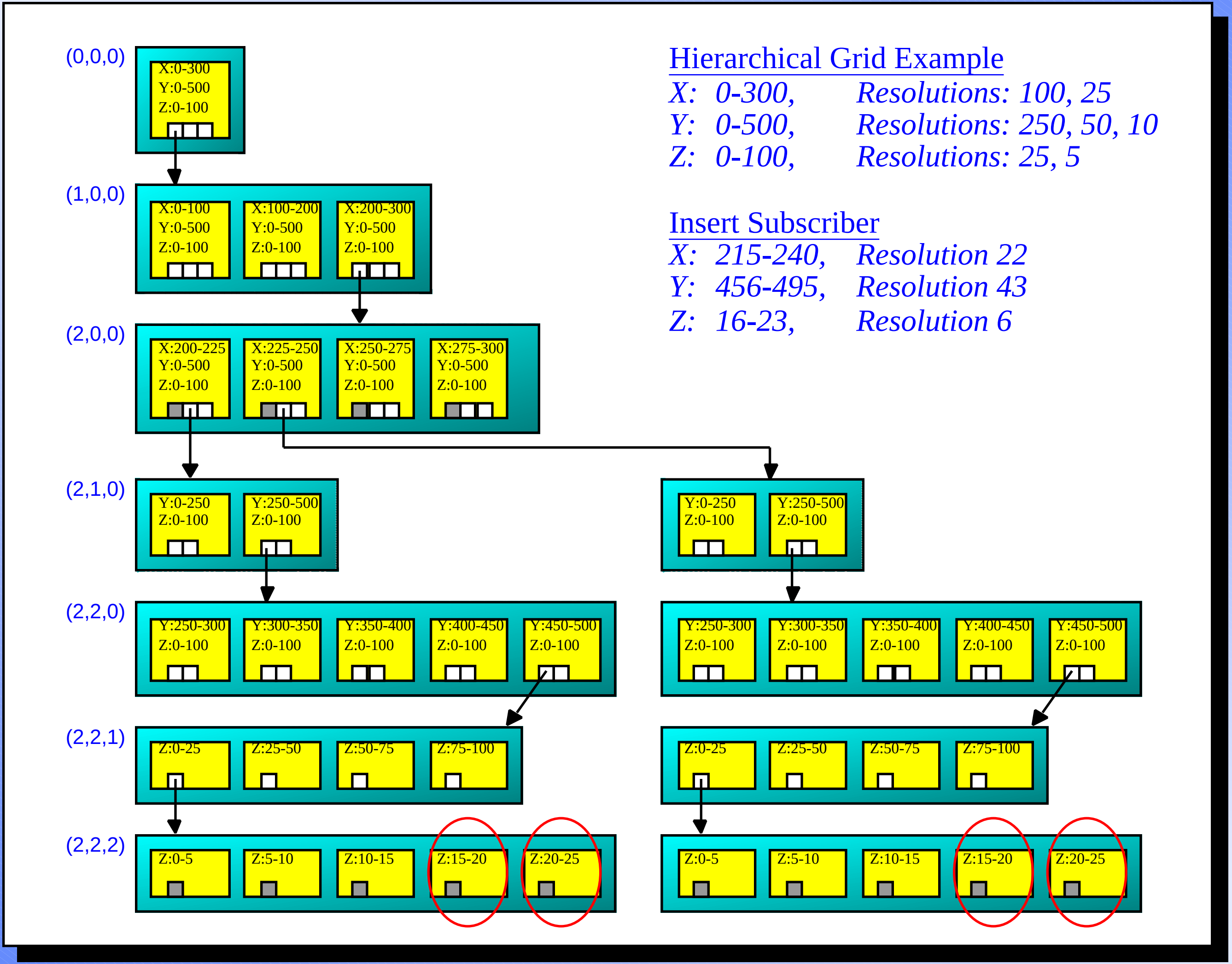
MORE ON GRIDS

- *In an Ideal DDM System, remote objects would appear when they are first detected and then disappear when they are no longer needed - i.e., efficient use of resources (Grids can help achieve this)*
  - *Storage of remote objects only when needed*
  - *Complicated computations performed only when necessary*
  - *Attribute update messages only sent when necessary*
- *Grid sizes must match update and subscription regions*
  - *Bad: Big sensors - little grids (too much overhead)*
  - *Bad: Little sensors - big grids (inefficient filtering)*
  - *Bad: Fast movers - little grids (too much overhead)*
  - *Bad: Slow movers - big grids (inefficient filtering)*
  - *Good: distributed hierarchical grids can help tremendously!*

WHERE GRIDS FAIL

- Potential for time-driven event scheduling
  - Checking in and out of grids while nothing else is going on
  - Space-based sensors that can see very large regions
  - Must choose a time increment that is so large that the effectiveness of the grid-based filtering is poor
- Need a fallback position when grids fail
  - Precomputation scheme on the receiver's side with message throttling protocol will provide additional filtering
  - Specifics of protocol are TBD (but we have some ideas)
    - Bookkeeping may get very complicated
    - May take some time to get right (will not let this affect schedule)
    - Long term optimization effort, not on critical path

HIERARCHICAL GRID EXAMPLE



Hierarchical Grid Example

X:	*0-300,*	*Resolutions: 100, 25*
Y:	*0-500,*	*Resolutions: 250, 50, 10*
Z:	*0-100,*	*Resolutions: 25, 5*

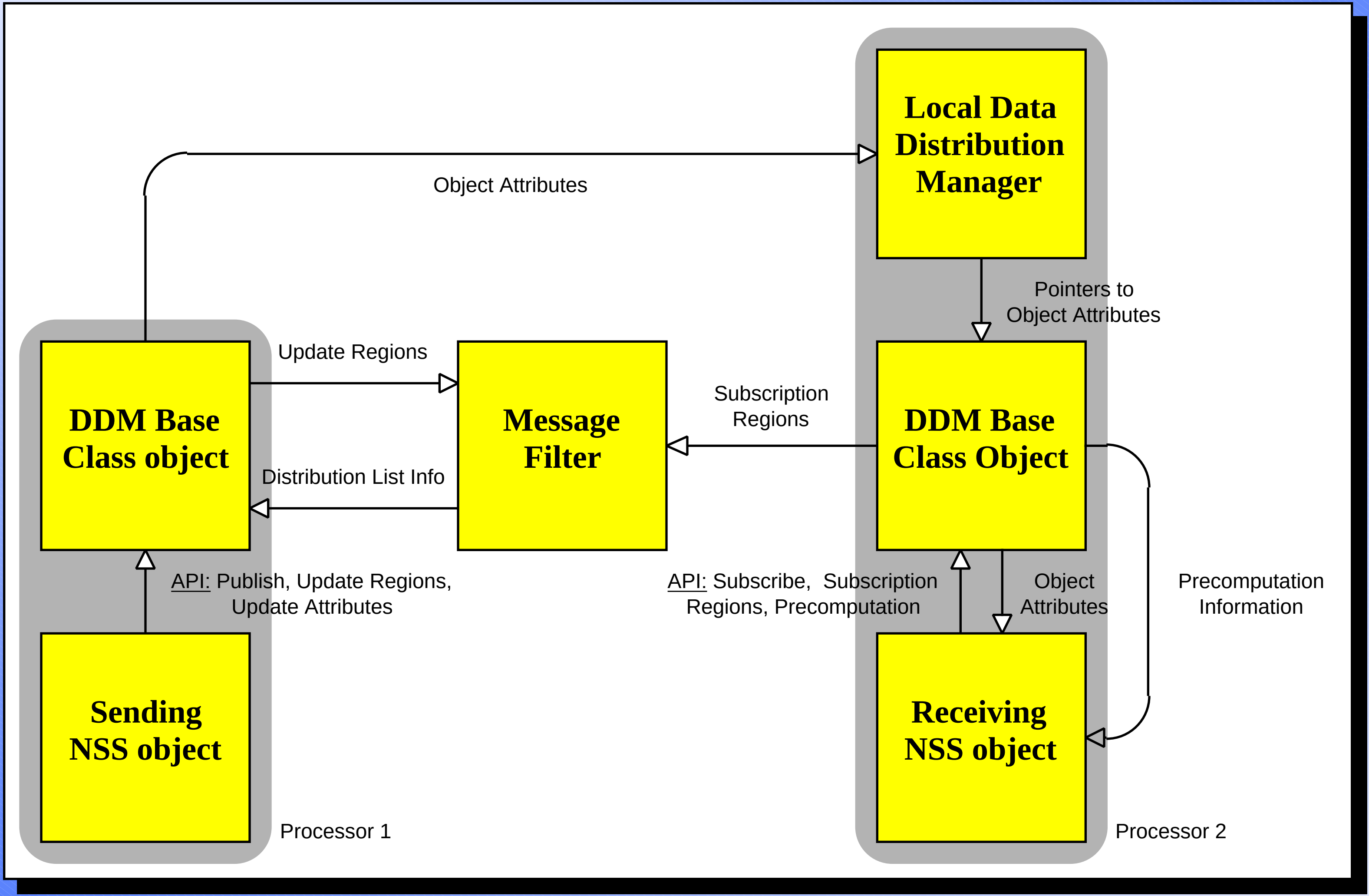
Insert Subscriber

X:	*215-240,*	*Resolution 22*
Y:	*456-495,*	*Resolution 43*
Z:	*16-23,*	*Resolution 6*

SPACES INPUT FILE

```
InterestSpaces {
    reference SPACE WWII
    reference SPACE WWIICom
}
WWII {
    reference THEATER Europe
    reference THEATER Pacific
    reference DIMENSION CrossSection
    reference ENUMTYPE Alliances
    reference CATEGORIES WWIICategories
}
Europe {
    LatLng {
        Latitude {
            float Lo 20
            float Hi 50
        }
        Longitude {
            float Lo -170
            float Hi -130
        }
        float Resolution[0] 521.2
        float Resolution[1] 154.3
        float Resolution[2] 35.8
        float Resolution[3] 3.7
        logical Distribute T
}
    Altitude {
        float Lo 0.0
        float Hi 10.0
        float Resolution[0] 3.5
        logical Distribute T
    }
}
Pacific {
    LatLng {
        Latitude {
            float Lo 10
            float Hi 50
        }
        Longitude {
            float Lo 125
            float Hi -150
        }
        float Resolution[0] 936.5
        float Resolution[1] 180.4
        float Resolution[2] 25.3
        float Resolution[3] 2.6
        logical Distribute T
    }
    Altitude {
        logical Distribute F
        float Resolution[0] 1.3
    }
}
CrossSection {
    float Lo 1.0
    float Hi 10.0
    float Resolution[0] 2.0
    float Resolution[1] 0.4
    logical Distribute F
}
ShipTypes {
    enum BattleShip
    enum AirCraftCarrier
    enum PtBoat
    logical Distribute F
}
Alliances {
    enum Allies
    enum Axis
    logical Distribute T
}
WWIICategories {
    int Ncategories 1000
    logical Distribute T
}
WWIICom {
    reference DIMENSION Frequency
    ENUMTYPE CommTypes {
        enum Radio
        enum Voice
        logical Distribute T
    }
}
Frequency {
    float Lo 2.0
    float Hi 3.0
    float Resolution[0] 1.06
    logical Distribute T
}
```

DATA DISTRIBUTION DESIGN



EXAMPLE OF FILTERING

- Aircraft1 changes its update region (or Aircraft2 changes its subscription region)
  - Messages sent to new grids and old (no longer needed) grids
- Messages from grids inform Aircraft1 that it is within Aircraft2's subscription region (it might be detected)
- Aircraft1 sends its object attributes to the LDDM on Aircraft2's node
- The LDDM hands a pointer of Aircraft1's attributes to Aircraft2
  - This activates Aircraft2's precomputation event
    - If detection is predicted, a wakeup event is scheduled for the time when Aircraft1 is first detected by Aircraft2 - Otherwise, the (TBD) message throttling filter protocol is activated

DATA DISTRIBUTION DESIGN

- *Main Components*
  - *Data Distribution Management (DDM) Base Class Object provides DDM services to NSS entity objects*
    - *Sending functionality*
      - *Allows an object to define its published attributes*
      - *Packages published attributes for distribution via its distribution list*
      - *Coordinates update regions used for filtering*
      - *Automates delivery of updated attributes*
    - *Receiving functionality*
      - *Allows an object to subscribe to attributes of other objects*
      - *Provides values of other object's attributes*
      - *Coordinates subscription regions used for filtering*
      - *Activates precomputations when new attributes are received*
    - *Base class is reusable for other military simulations*
    - *Can provide HLA functionality for conservative federates*

DATA DISTRIBUTION DESIGN

- *Main Components - Continued...*
  - *Message Filter - computational and message scalability*
    - *Provides sending and receiving interfaces for object attributes*
    - *Establishes a distribution list for each object indicating which other objects need its attributes*
    - *Must be distributed (no central hot spots)*
  - *Precomputation Filter - computational scalability*
    - *Uses received object attributes to further eliminate interactions or to compute when they might occur*
    - *Computations are performed by the receiving objects which are inherently distributed*
  - *Local Data Distribution Manager (LDDM) - memory scalability*
    - *Manages the distribution of received object attributes*
    - *Maintains a single copy of the attributes (distributes pointers)*

BASE CLASS DDM OBJECT

- Published Attributes
  - Attribute Objects (C++ Objects)
  - Dynamic Attributes (List)
  - Motion (Library of Motion Objects in List)
  - Volumes
- Subscribed Attributes
  - Class Types
    - Attribute Names for Each Class
- Update Regions
  - Ties Expanded Published Attributes to Interest Spaces
- Subscription Regions
  - Ties Expanded Published Attributes to Interest Spaces
- Precomputation Information
  - General Mechanism to Specify Precomputation
  - Wake-Up Events when Precomputation passes
  - Receiver-Initiated Supression (Protocol TBD)
- List of Grids Used in Update Regions
- List of Grids Used in Subscription Regions
- Distribution List of Other Objects Needing Published Attributes
- Sensor Reception List of Multicast Groups (TBD)
- Pointers to Attributes of Reflected Objects

HIERARCHICAL GRIDS

List of Dimensions in Interest Space
- Lo and Hi Ranges for Each Dimension
- List of Resolutions for Each Dimension
- Children for Each Dimension (if Needed)

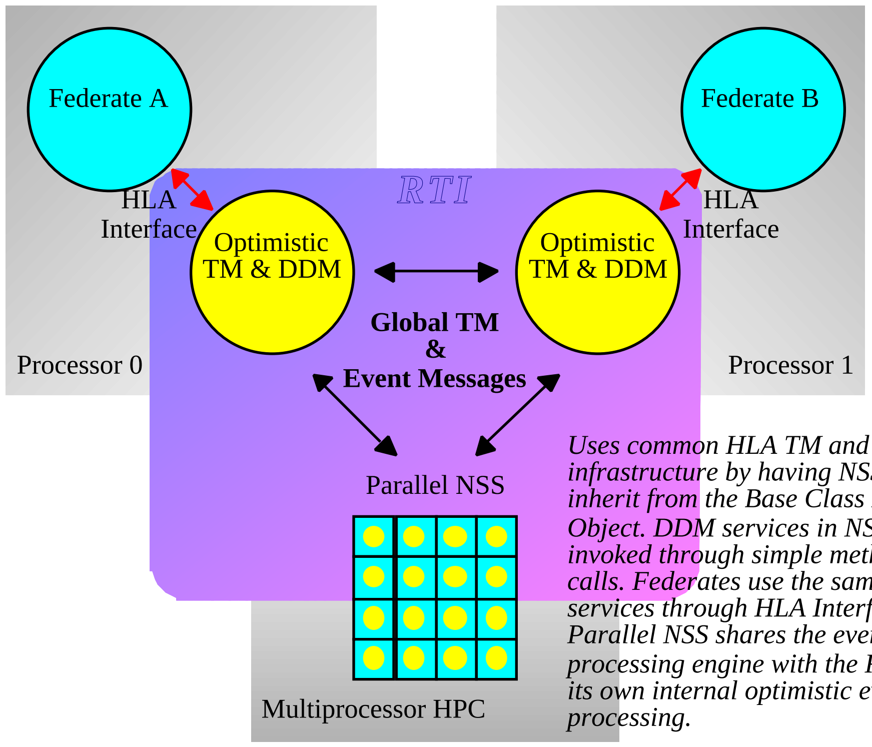
Lists of Subscribers and Publishers in Grids
Hash Table for Publishers and Subscribers for Fast Lookups
Free List of Items for the Publisher and Subscriber Lists

Note:
- ~100 Per Node (One Would be a Bottleneck)
- "Space" Object Manager Creates the Hierarchical Grids and Distributes them According to an Input File
- Physical Space is handled as a Special Case using a Multi-Resolution Sperical Earth with different altitudes

LOCAL DATA DISTRIBUTION MANAGER

List of Reflected Objects
- Set of Attribute Values for Each Object
- List of Local Receiving Objects
- Receiver-Initiated Supression Information (TBD)

RTI & FEDERATES

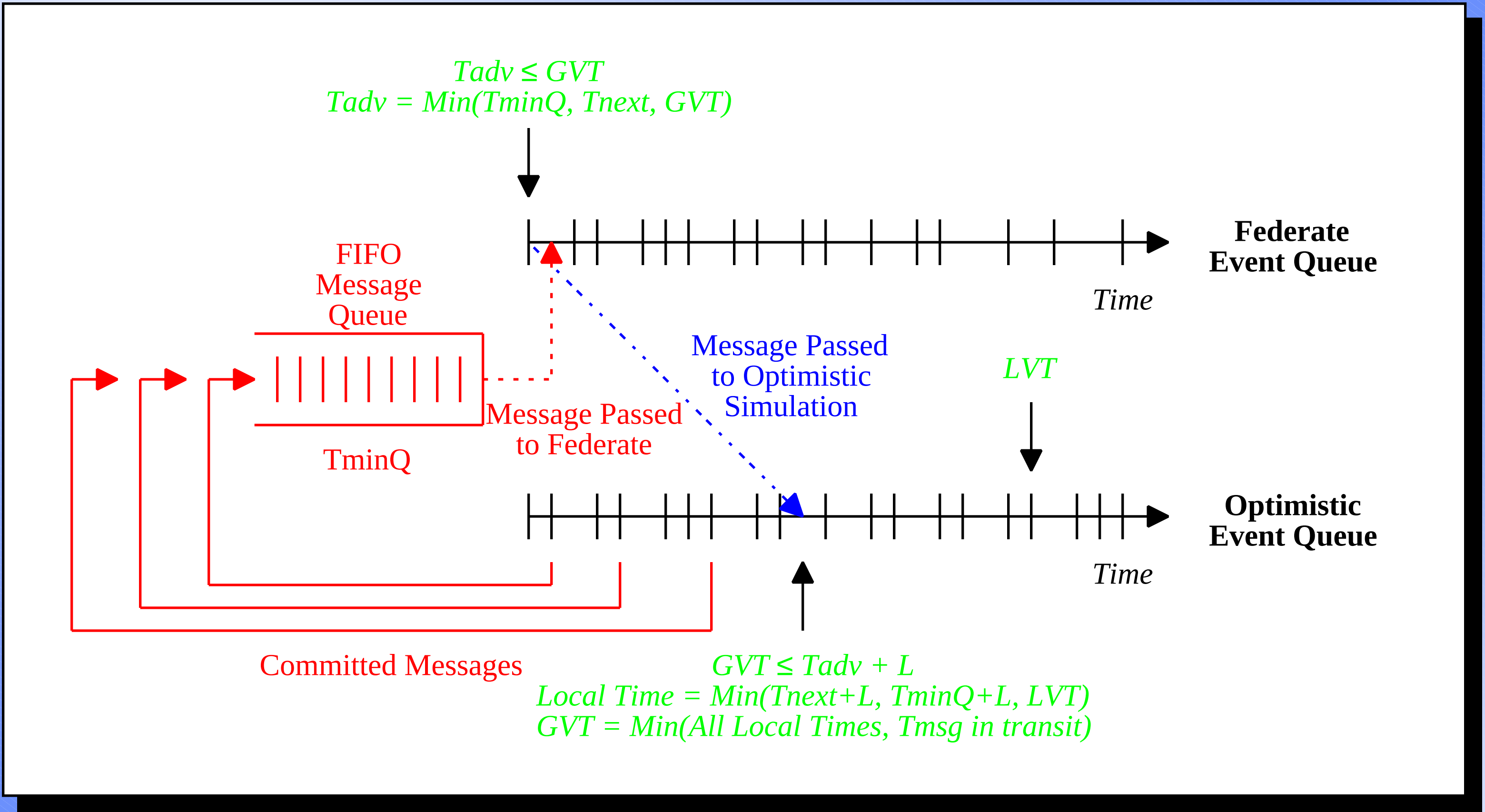


Uses common HLA TM and DDM infrastructure by having NSS entities inherit from the Base Class DDM Object. DDM services in NSS are invoked through simple method calls. Federates use the same services through HLA Interfaces. Parallel NSS shares the event-processing engine with the RTI for its own internal optimistic event processing.

Can Coordinate Between Federate and Optimistic RTI in Three Ways:

- Single Threaded With Federate in Control
- Multiple Processes Communicating Through Shared Memory and Signals
- Multi-Threaded (One UNIX Process, Multiple Light-Weight Processes)

TIME MANAGEMENT & HLA


Tadv ≤ GVT
Tadv = Min(TminQ, Tnext, GVT)
FIFO Message Queue
TminQ
Message Passed to Federate
Message Passed to Optimistic Simulation
Federate Event Queue
Time
LVT
Optimistic Event Queue
Time
Committed Messages
GVT ≤ Tadv + L
Local Time = Min(Tnext+L, TminQ+L, LVT)
GVT = Min(All Local Times, Tmsg in transit)

MISCELLANEOUS ONGOING TASKS

- *Incremental State Saving*
  - *Support for object-oriented stream I/O*
  - *Barriers for throttling GVT updates*
  - *Extensible Interface so applications can define their own rollbackable operations*
  - *Rollback wrappers for Rogue Wave Tools*
- *Porting Efforts*
  - *Distributed Workstations (Socket Bug Fixed) HP, SUN, SGI*
  - *Convex Exemplar (Standard UNIX Shared Memory services)*
  - *MPI work being done by Dartmouth*
  - *Paragon, SP2, & virtually all HPC machines*

SUMMARY

- *Rehosting NSS to run in parallel using SPEEDES to support much faster than real time strike plan evaluation using plans generated by APPEX*
- *The starting point, and most critical step is developing the data distribution management function*
  - *Needed to cleanly and efficiently distribute NSS*
  - *Has potential reuse for other distributed military simulations*
  - *Will support near-term HLA prototype experiments*
  - *Data distribution management design based on experience from SPEEDES and NSS*
- *Achievable objectives*
  - *Technologically feasible*